

Inside Android's UI

Embedded Linux Conference Europe 2012

Karim Yaghmour
@karimyaghmour

karim.yaghmour@opersys.com





These slides are made available to you under a Creative Commons Share-Alike 3.0 license. The full terms of this license are here:
<https://creativecommons.org/licenses/by-sa/3.0/>

Attribution requirements and misc., PLEASE READ:

- This slide must remain as-is in this specific location (slide #2), everything else you are free to change; including the logo :-)
- Use of figures in other documents must feature the below “Originals at” URL immediately under that figure and the below copyright notice where appropriate.
- You are free to fill in the “Delivered and/or customized by” space on the right as you see fit.
- You are FORBIDDEN from using the default “About” slide as-is or any of its contents.
- You are FORBIDDEN from using any content provided by 3rd parties without the EXPLICIT consent from those parties.

(C) Copyright 2012, Opersys inc.

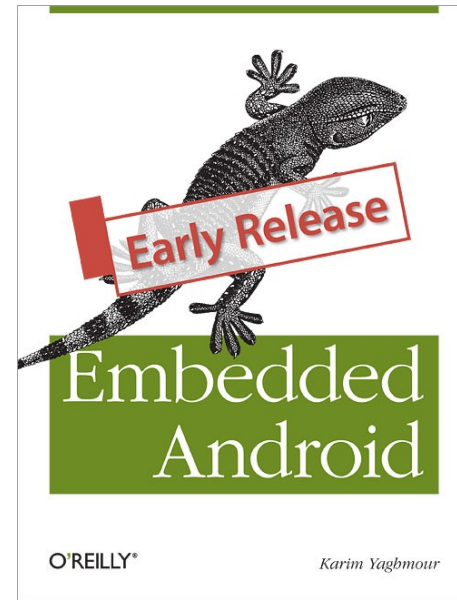
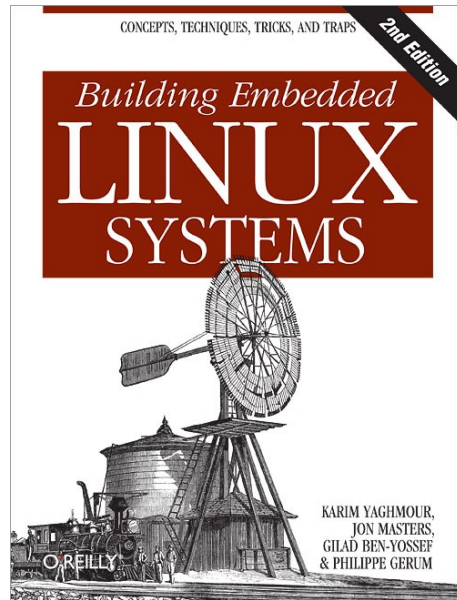
These slides created by: Karim Yaghmour

Originals at: www.opersys.com/community/docs

Delivered and/or customized by

About

- Author of:

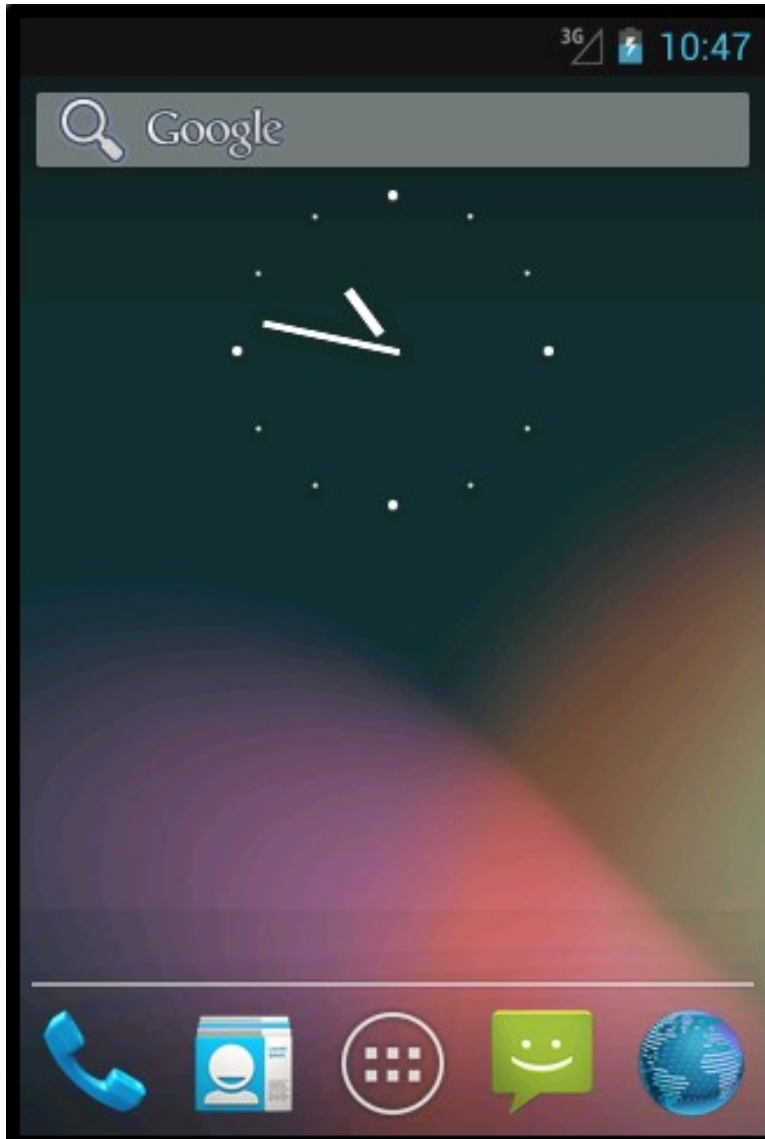


- Introduced Linux Trace Toolkit in 1999
- Originated Adeos and relayfs (kernel/relay.c)
- Training, Custom Dev, Consulting, ...

Agenda

- Android's UI, what's that?
- Architecture Basics
- Display Core
- OpenGL
- Input Layer
- Relevant Apps and Services
- System Startup
- References and Pointers

1. Android's UI, what's that?



???



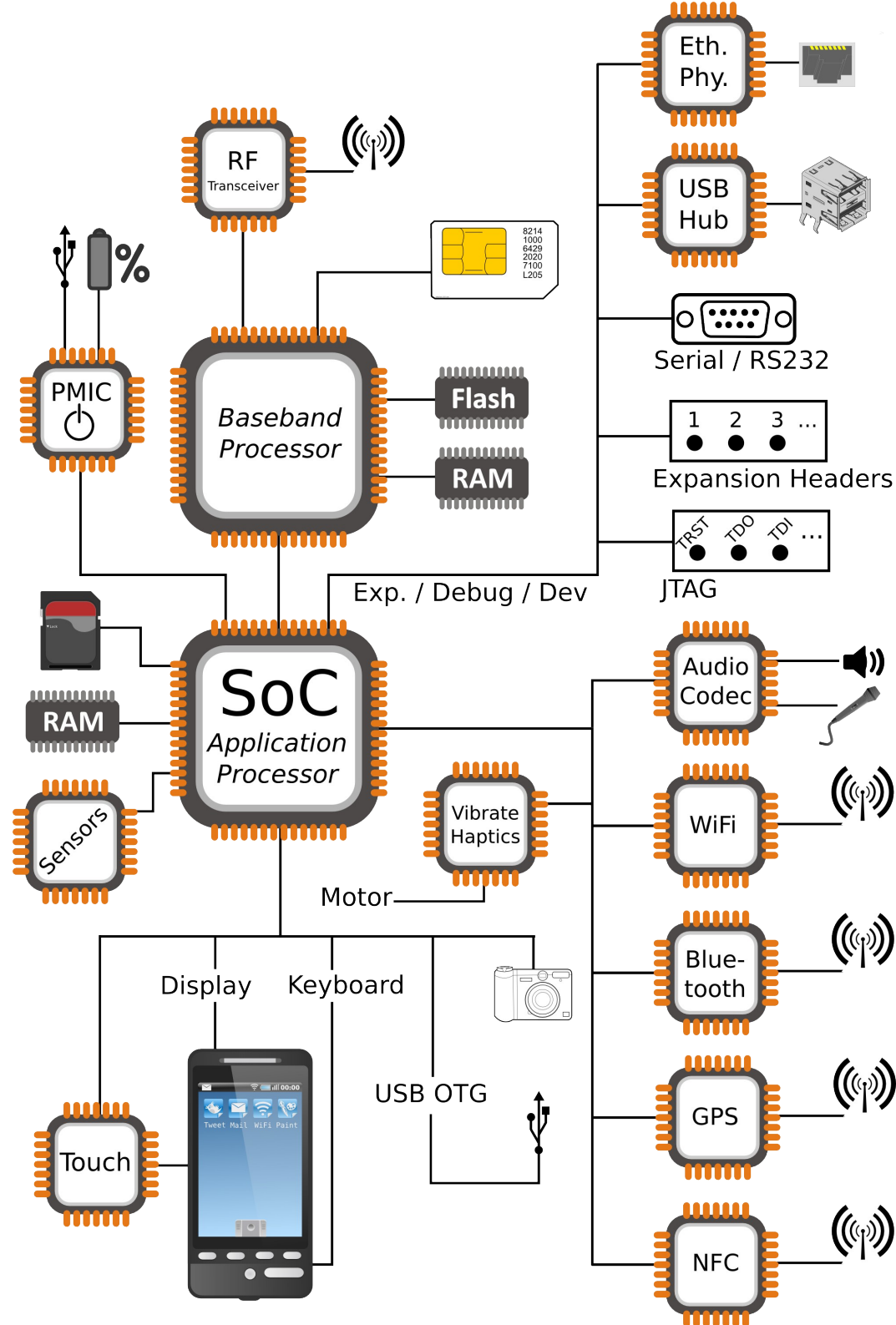
- SoC / GPU
- Touch input
- LCD
- Keyboard

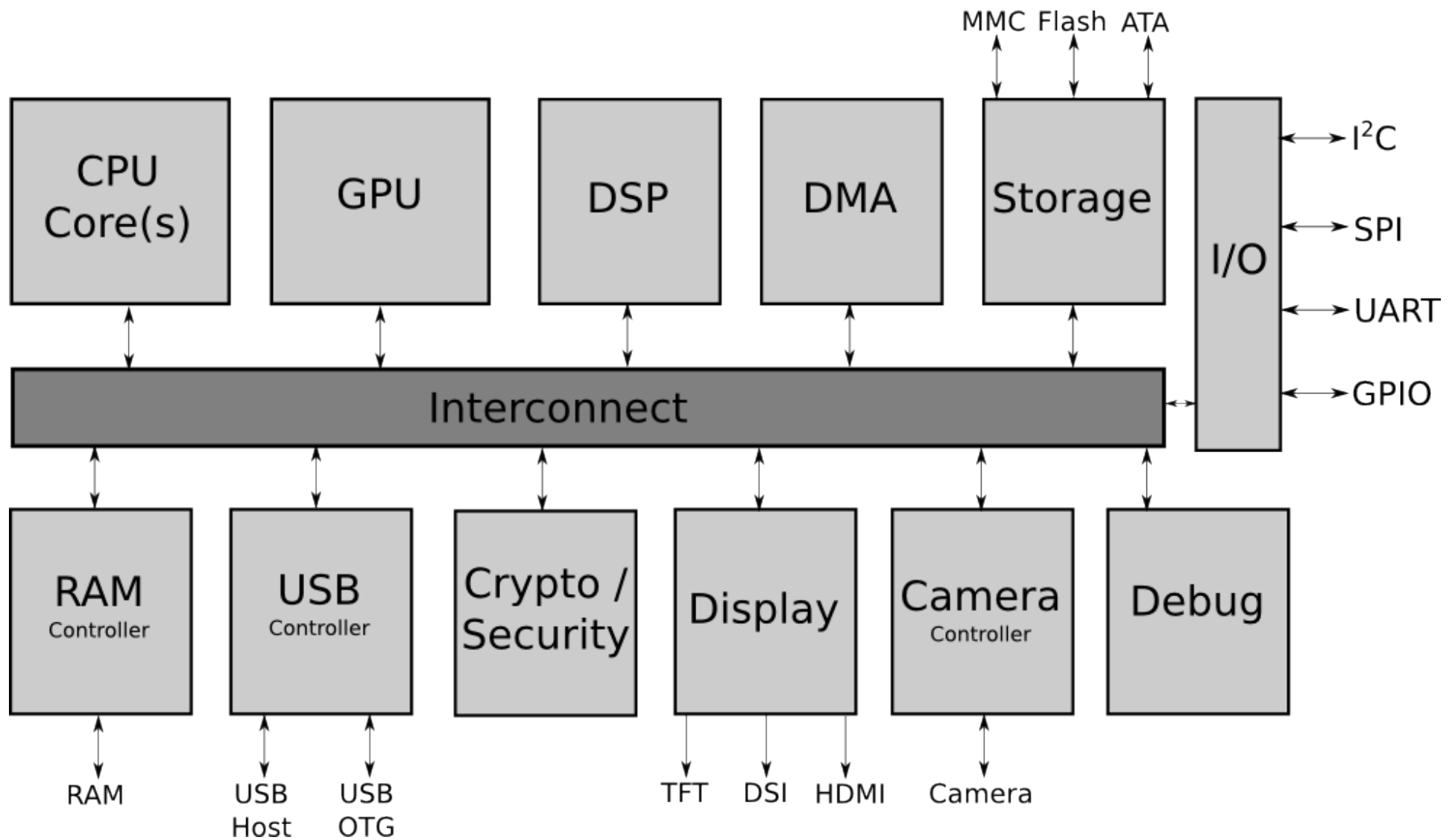
1.1. What I'm NOT covering

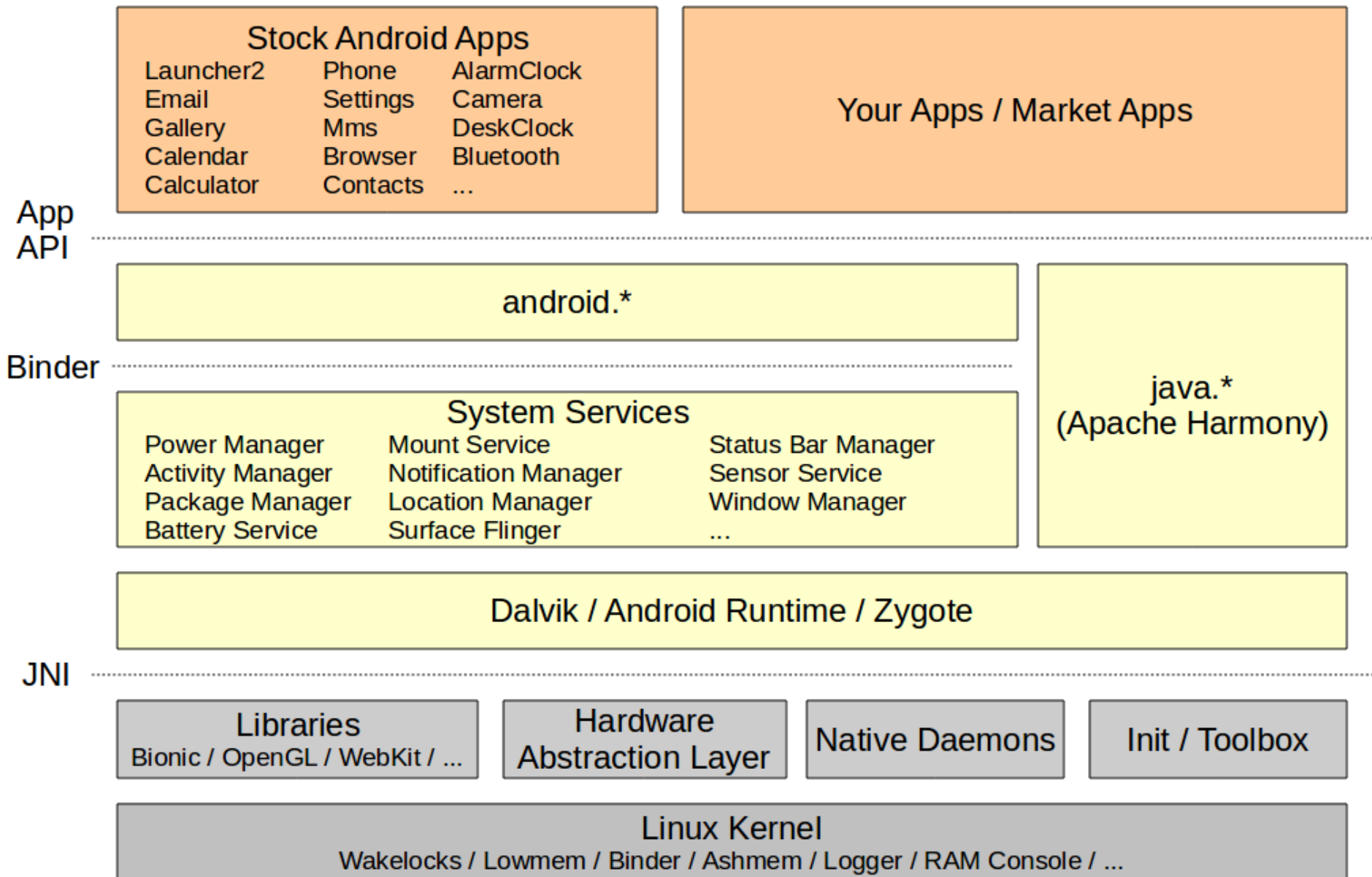
- Media layer
- StageFright
- Video playback
- Camera
- DRM
- Etc.

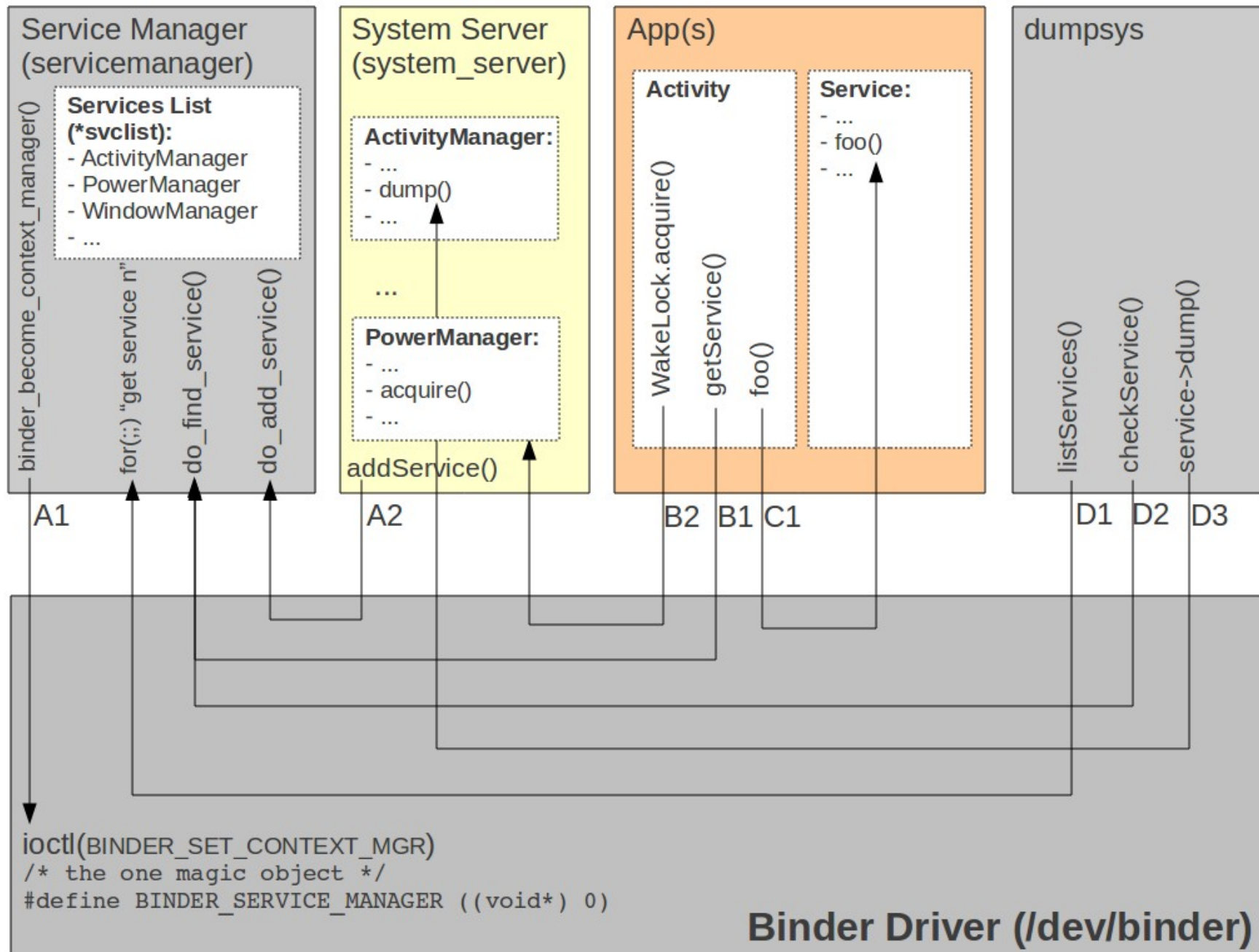
2. Architecture Basics

- Hardware used to run Android
- AOSP
- Binder
- System Services
- HAL









System Services

System Server

Java-built Services

Power Manager	Mount Service
Activity Manager	Notification Manager
Package Manager	Location Manager
Battery Service	Search Service
Window Manager	Wallpaper Service
Status Bar	Headset Observer
Clipboard Service	...

C-built Services

Sensor Service

Surface Flinger

Media Service

Audio Flinger
Media Player Service
Camera Service
Audio Policy Service

Includes:

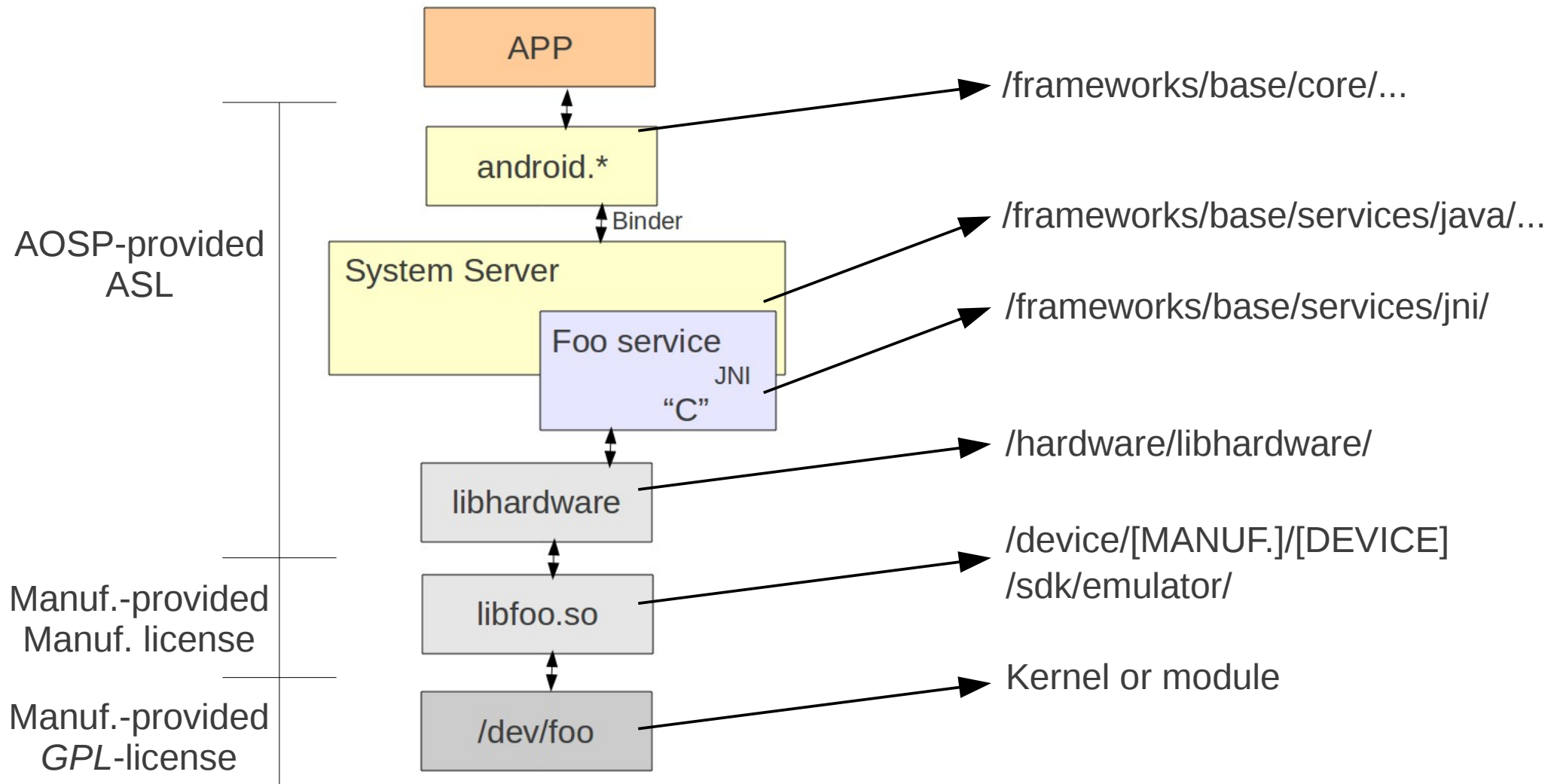
- StageFright
- Audio effects
- DRM framework

Phone App

JNI

Native Methods for
Java-built Services

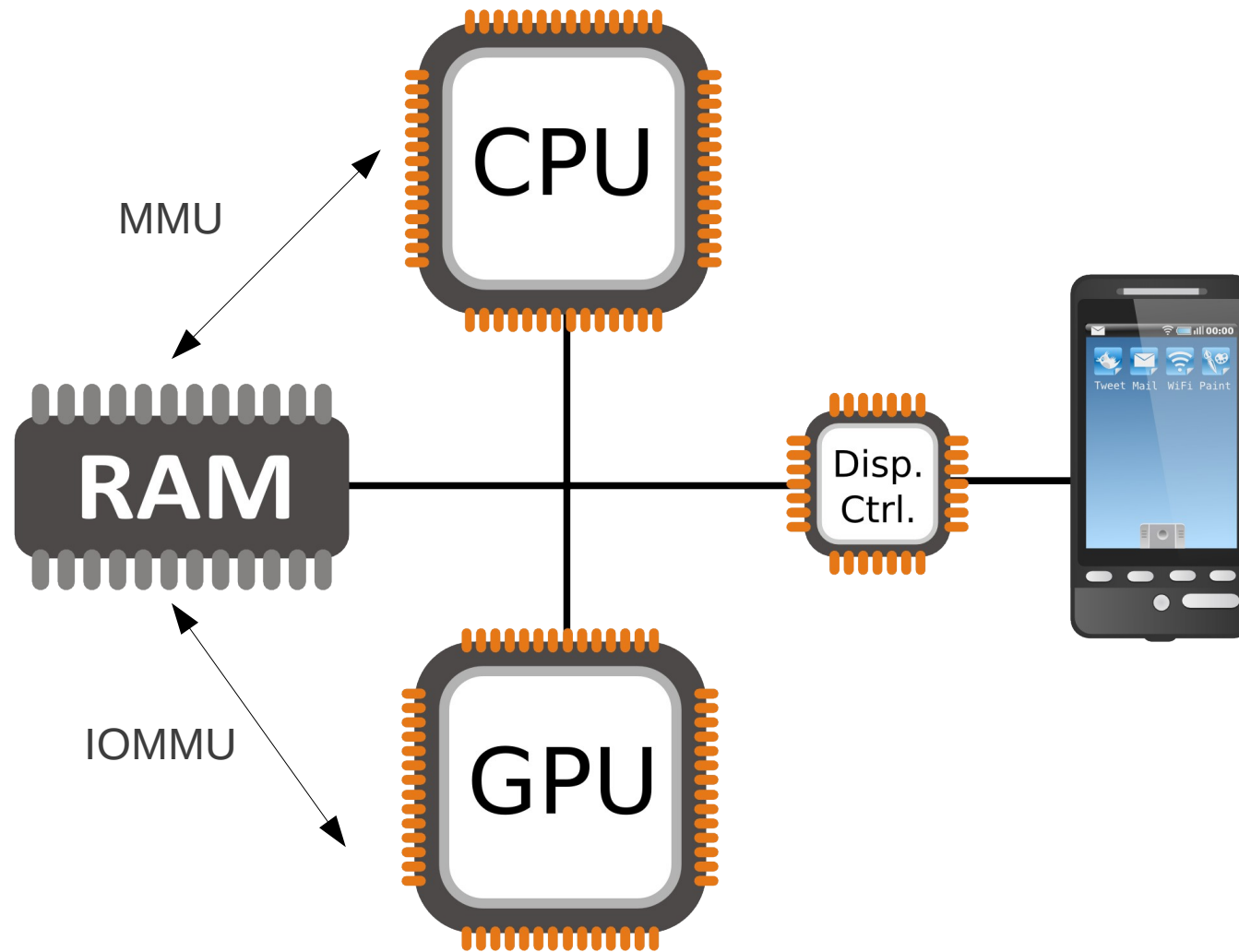
Hardware Abstraction Layer



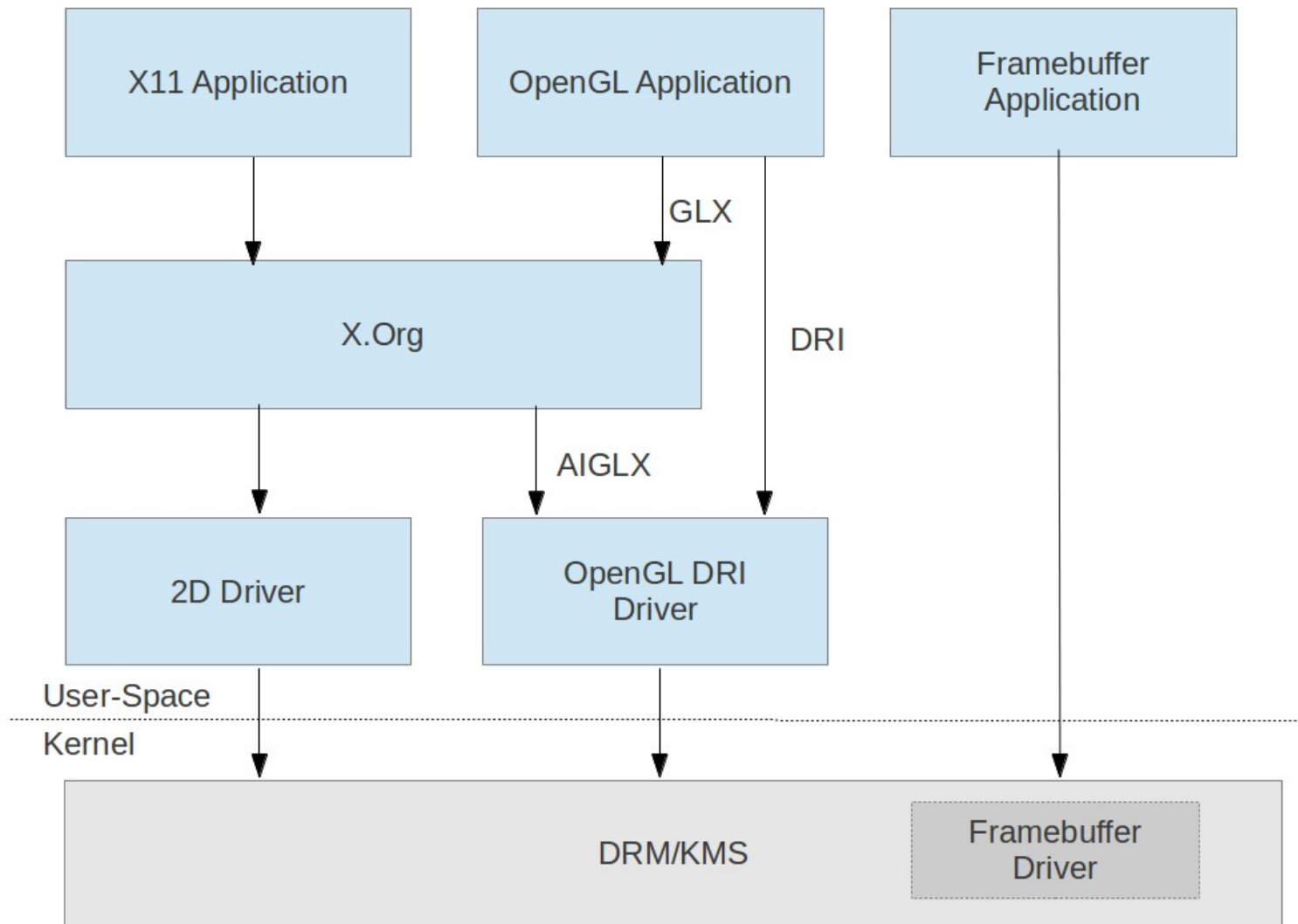
3. Display Core

- Display Hardware
- Classic Linux display stack
- Display stack in Android
- Kernel driver
- HAL definition
- HAL module
- Surface Flinger
- Window Manager
- Walkthrough

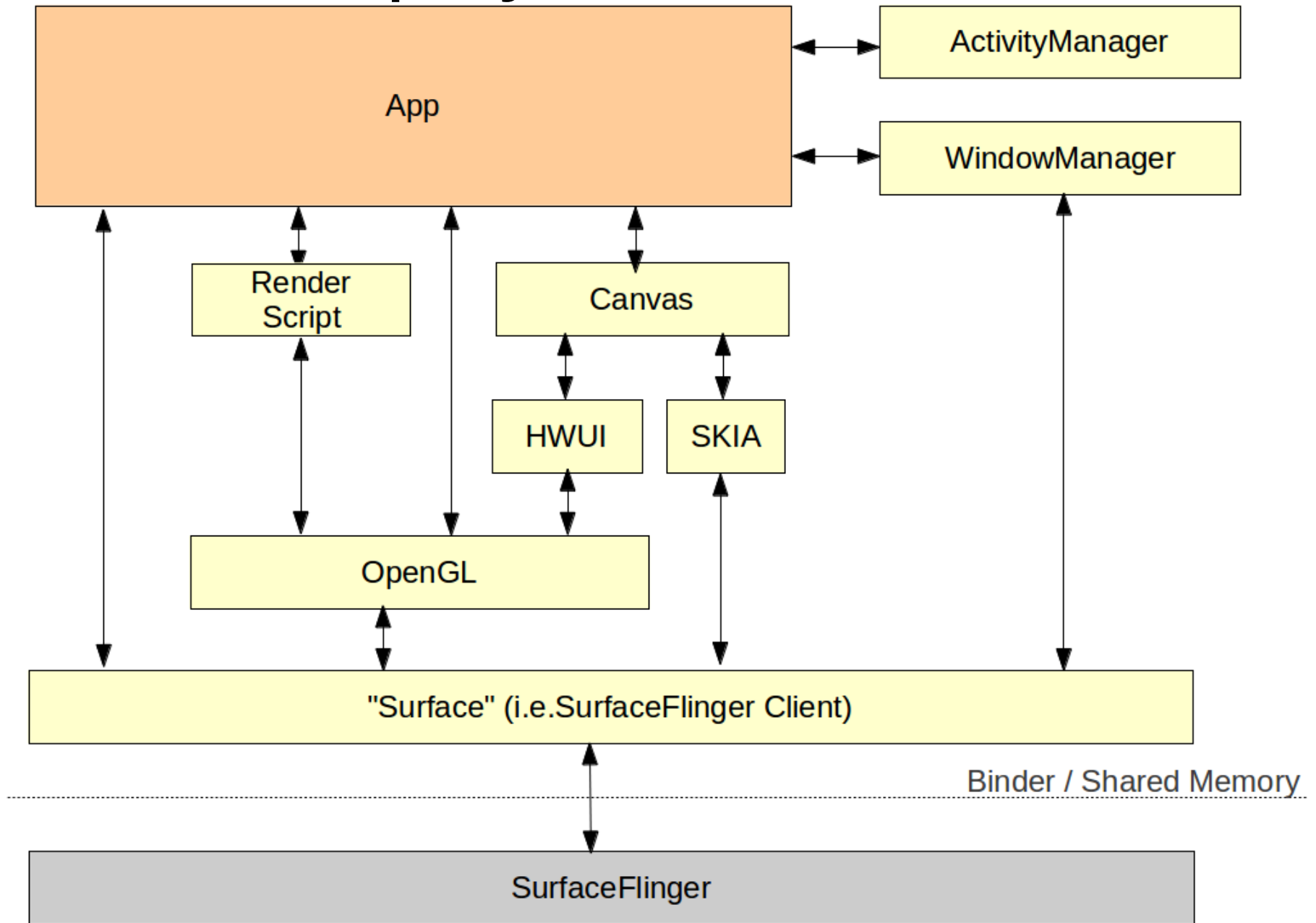
3.1. Display Hardware

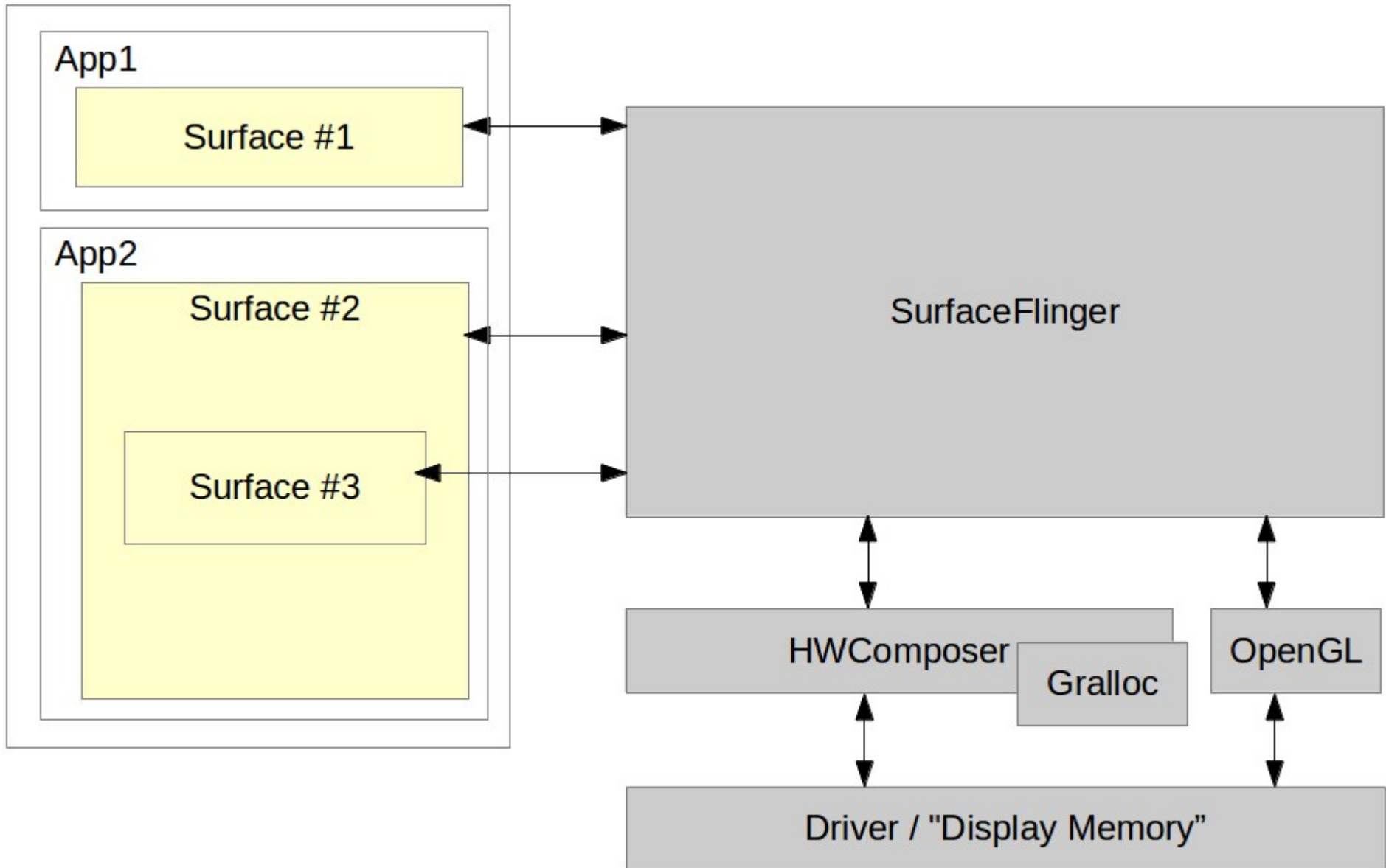


3.2. Classic Linux display stack



3.3. Display stack in Android





3.4. Kernel driver

- Video memory management
- Mode setting
- Checking of parameters
- Motorola Xoom:
 - /dev/nvhdcpl
 - /dev/nvhost-ctrl
 - /dev/nvhost-display
 - /dev/nvhost-dsi
 - /dev/nvhost-gr2d
 - /dev/nvhost-gr3d
 - /dev/nvhost-isp
 - /dev/nvhost-mpe
 - /dev/nvhost-vi
 - /dev/nvmap
 - /dev/tegra-crypto
 - /dev/tegra_avp
 - /dev/tegra_camera
 - /dev/tegra_fuse
 - /dev/tegra_rpc
 - /dev/tegra_sema
- ... whatever hides in hwcomposer HAL module

3.5. HAL Definition

- `/hardware/libhardware/include/hardware/hwcomposer.h`
- `struct hwc_procs`:
 - `invalidate()`
 - `vsync()`
- `struct hwc_composer_device`:
 - `prepare()`
 - `set()`
 - `dump()`
 - `registerProcs()`
 - `query()`
 - `*()`

3.6. HAL module

- Skeleton `/hardware/libhardware/modules/hwcomposer.cpp`
- `/system/lib/hw/hwcomposer.BOARD.so`
- `/system/lib/hw/gralloc.BOARD.so`
- Ex. - Mot Xoom:
 - `hwcomposer.tegra.so`
 - `gralloc.tegra.so`
- Surface Flinger hook:
 - `/frameworks/native/services/surfaceflinger/DisplayHardware`
 - `HWComposer.cpp`
 - Provides fake vsync if none is provided in HW

3.7. Surface Flinger

- Actual server:
 - /frameworks/native/services/surfaceflinger
- Client side:
 - /frameworks/native/libs/gui
- Client / Server interface:
 - ISurfaceComposerClient.cpp
 - ISurfaceComposer.cpp
- This is NOT an aidl'ed service
- All communication is manually marshalled/unmarshalled

3.8. Window Manager

- Server side:
 - /frameworks/base/services/java/com/android/server/wm/
 - WindowManagerService.java
 - Session.java
- Client side:
 - /frameworks/base/core/java/android/view/
 - WindowManager.java
 - WindowManagerImpl.java
 - ViewRootImpl.java
- Interfaces:
 - IWindowManager.aidl
 - IWindowSession.aidl
- Parameters (incl. z-order):
 - See WindowManager.java

3.9. Walkthrough

- Activity Manager relies on Activity Thread
- AT calls on `attach()` and `makeVisible()`
- `makeVisible` does `wm.addView()`
- `wm.addView()` - this also called by `StatusBar` to display itself
 - Creates a new `ViewRootImpl`
 - Call on its `setView()`
- `setView()` calls on `sWindowSession.add(...)`
- This results in call to WM's `addWindow()`
- `ViewRootImpl`'s `performTraversals()`
 - Calls on `layoutWindow()`
 - Calls to WM session's `layout()`
 - Call to WM's `layoutWindow()`
 - Call to `createSurfaceLocked()`
 - `new Surface(...)`

4. OpenGL

- What's OpenGL?
- What's in a modern-day GPU?
- Software layers involved
 - Kernel driver
 - EGL libs
 - Native interface
 - Java interface
 - Software GL implementation

4.1. What's OpenGL?

- It's just an API ... nothing but an API ...
- Check out Wikipedia
- Multiple versions out
- “ES” versions for embedded use
- Up to ES 3.0
- Android support up to ES 2.0

4.2. What's in a modern-day GPU?

- A tremendous amount of parallel processing units
- “SIMD”-like instruction set
- Video decoding/encoding capabilities
- ...

4.3. Software layers involved

- Kernel driver
- GL libraries
- Native GL API
- Java GL API

4.4. Kernel driver

PROPRIETARY

4.5. EGL libs

- /frameworks/base/native/opengl/libs
- Entry point: /system/lib/libEGL.so
- Looks for /system/lib/egl/egl.cfg
- /system/lib/egl - Mot Xoom:
 - egl.cfg
 - libEGL_perfhud.so
 - libEGL_tegra.so
 - libGLES_android.so
 - libGLESv1_CM_perfhud.so
 - libGLESv1_CM_tegra.so
 - libGLESv2_perfhud.so
 - libGLESv2_tegra.so
- elg.cfg:

0 0 tegra

4.6. Native interface

- /frameworks/native/opengl/include
 - EGL
 - ETC1
 - GLES
 - GLES2
 - KHR

4.7. Java interface

- GL libs required by libandroid_runtime.so
- /frameworks/base/opengl/java/android/opengl:
 - ...

4.8. Software GL implementation

- `/frameworks/native/opengl/libagl`

5. Input Layer

- Kernel side - “std” Linux input layer:
 - /dev/input/*
- No HAL use
- Native lib:
 - libinput
 - /frameworks/base/services/input
- Input Manager Service:
 - /frameworks/base/services/java/com/android/server/input
 - Started and directly tied to Window Manager
- Specific config files (see source.android.com)
- Soft keyboard:
 - /frameworks/base/core/java/android/inputmethodservice
- Input methods:
 - /packages/inputmehods
 - <http://developer.android.com/guide/topics/text/creating-input-method.html>

6. Relevant Apps and Services

- Launcher
- StatusBar
- Wallpaper Manager Service
- Notification Service
- App Widgets

6.1. Launcher

- An app like any other
- See `/packages/app/Launcher2`

6.2. StatusBar

- A unique app
- See `/frameworks/base/packages/SystemUI`
- Connects to Status Bar Manager and gives an interface it can use to call back into Status Bar
- Can use `setIcon()` to display icons on the right
- Provides a CPU usage add-on that renders straight on rest of display using higher z-order

6.3. Wallpaper Manager Service

- See
`/frameworks/base/services/java/com/android/server/WallpaperManagerService.java`

6.4. Notification Service

- Toasts
- Status bar notifications
- Gets handle to Status Bar Service at instantiation
- Uses handle to communicate with Status Bar

6.5. App Widgets

- See
`/frameworks/base/services/java/com/android/server/AppWidgetService.java`

7. System Startup

- Kernel
- Init
- Boot animation
- Launcher

7.1. Boot animation

- Started by Surface Flinger
- “bootanim” binary
- /frameworks/base/cmds/bootanimation
- Relies on bootanimation.zip w/ PNGs (nothing but)
- See
https://github.com/CyanogenMod/android_vendor_cm/tree/jellybean/prebuilt/common/bootanimatino
- Must contain a desc.txt:
 <width> <height> <fps>
 p <count> <pause> <path>
 p <count> <pause> <path>

8. References and Pointers

- “Use the source, Luke”
- Jim Huang's “Android Graphics”
- Benjamin Zores' “Linux Magazine / France” articles
- MIPS article on graphics internals:
- <http://developer.mips.com/2012/04/11/learning-about-android-graphics-subsystem/>
- Stéphane Marchesin's “Linux Graphics Drivers: an Introduction”
- <http://source.android.com/tech/input/index.html>

Thank you ...

karim.yaghmour@opersys.com

